An experimental survey on big data frameworks (Highlight paper)

Wissem Inoubli
inoubliwissem@gmail.com
University of Tunis El Manar, Faculty of sciences of Tunis, LIPAH, 1060, Tunis, Tunisia

Sabeur Aridhi
sabeur.aridhi@loria.fr
University of Lorraine, LORIA, Campus Scientifique BP 239, Vandoeuvre-lès-Nancy, France

Haithem Mezni
haithem.mezni@fsjegj.rnu.tn
University of Jendouba, Avenue de l 'Union du Maghreb Arabe, Jendouba 8189, Tunisia

Mondher Maddouri
maddourimondher@yahoo.fr
College Of Business, University of Jeddah, P.O.Box 80327, Jeddah 21589, Saudi Arabia

Engelbert Mephu Nguifo
mephu@isima.fr
University Clermont Auvergne, CNRS, LIMOS, F-63000 Clermont-Ferrand, France

ABSTRACT

Recently, increasingly large amounts of data are generated from a variety of sources. Existing data processing technologies are not suitable to cope with the huge amounts of generated data. Yet, many research works focus on Big Data, a buzzword referring to the processing of massive volumes of (unstructured) data. Recently proposed frameworks for Big Data applications help to store, analyze and process the data. In this paper, we discuss the challenges of Big Data and we survey existing Big Data frameworks. We also present an experimental evaluation and a comparative study of the most popular Big Data frameworks with several representative batch.

CCS CONCEPTS

• **Information systems** → *Database management system engines.*

KEYWORDS

Big data, MapReduce, Hadoop, HDFS, Spark, Flink, Storm, Samza, Batch/stream processing

1 INTRODUCTION

In recent decades, increasingly large amounts of data are generated from a variety of sources. The size of generated data per day on the Internet has already exceeded two exabytes [1]. Within one minute, 72 h of videos are uploaded to Youtube, around 30.000 new posts are created on the Tumblr blog platform, more than 100.000 Tweets are shared on Twitter and more than 200.000 pictures are posted on Facebook [1]. Big Data problems lead to several research questions such as (1) how to design scalable environments, (2) how to provide fault tolerance and (3) how to design efficient solutions. Most existing tools for storage, processing and analysis of data are inadequate for massive volumes of heterogeneous data. Consequently, there is an urgent need for more advanced and adequate Big Data solutions. Many definitions of Big Data have been proposed throughout the literature. Most of them agreed that Big Data problems share four main characteristics, referred to as the four V's (Volume, Variety, Veracity and Velocity) [3]. In this paper, we first give an overview of most popular and widely used Big Data frameworks which are designed to cope with the above mentioned Big Data problems. We identify some key features which characterize Big Data frameworks. Then, we present an experimental study on Big Data processing systems with several representative batch stream and iterative workloads.

2 CATEGORIZATION OF BIG DATA FRAMEWORKS

We present in this section a categorization of the presented frameworks according to data format, processing mode, used data sources, programming model, supported programming languages and cluster manager (for more details, please see [2]). As explained in [2], Hadoop, Flink and Storm use the key-value format to represent their data. This is motivated by the fact that the key-value format allows access to heterogeneous data. For Spark, both RDD and key-value models are used to allow fast data access. We have also classified the studied big data frameworks into two categories: (1) batch mode and (2) stream mode. Additional, Hadoop processes the data in batch mode, whereas the other frameworks allow the stream processing mode. In terms of physical architecture, we notice tha all the studied frameworks are deployed in a cluster architecture, and each framework uses a specified cluster manager. We note that most of the studied frameworks use YARN as cluster manager.

3 EXPERIMETAL EVALUATION OF BIG DATA FRAMEWORKS

We have performed an extensive set of experiments to highlight the strengths and weaknesses of popular Big Data frameworks. We provide more informatiosn about our experiments and the obtained results in [2] and in the following link: https://members.loria.fr/SAridhi/files/software/bigdata/.

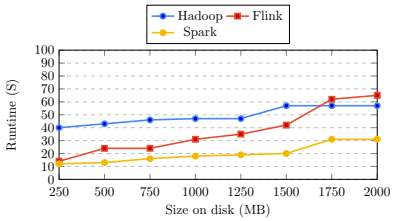

Figure 1: Impact of the size of the data on the average processing time

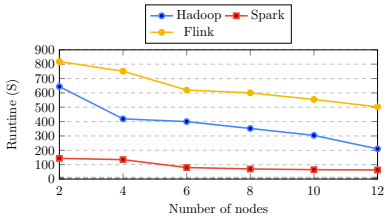

Figure 2: Impact of the number of machines on the average processing time (WordCount workload)

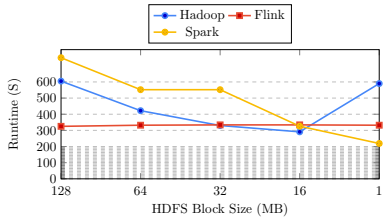

Figure 3: Impact of HDFS block size on the runtime (Kmeans workload with 10 million examples and 10 iterations)

3.1 Experimental results: Batch mode

This experiment aims to evaluate the impact of the size of the data on the processing time. The experiments are conducted using the WordCount workload and a set of text files with a size on disk varying from one GB to 100 GB. As shown in Fig. 1, Spark is the fastest framework in the case of small datasets, Flink is the next and Hadoop is the lowest, and Spark has kept its order in the case of big datasets and Hadoop showed good results compared to Flink (see [2]). In the next experiment, we tried to evaluate the scalability and the processing time of the considered frameworks based on the number of machines in the cluster. Fig. 2 shows that both Hadoop and Flink take higher time regardless of the cluster size, compared to Spark. In the next experiment, we try to show the impact of data partitioning on the studied frameworks. In our experimental setup, we used HDFS for storage. We varied the block size in our HDFS system and we run Kmeans with 10 iterations with all the used frameworks. Fig. 3 presents the impact of the HDFS block size on the processing time. As shown in Fig. 3, the curves are inflated proportionally to the size of the HDFS block size for both Hadoop and Spark while Flink does not imply any variation in the processing time.

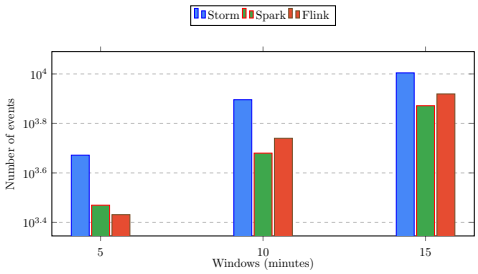

Figure 4: Impact of the window time on the number of processed events (100 KB per message)

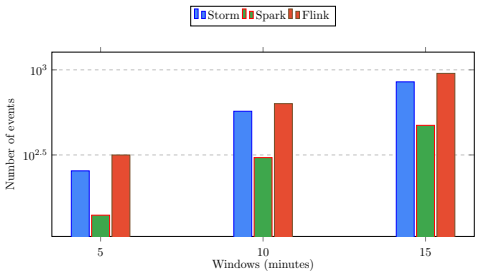

Figure 5: Impact of the window time on the number of processed events (500 KB per message)

3.2 Experimental results: Stream mode

The goal here is to compare the performance of the studied frameworks according to the number of processed messages within a period of time. In the first experiment, we send a tweet of 100 KB (in average) per message. The obtained results in Fig 4 show that Flink, Samza and Storm have better processing rates compared to Spark.

In the second experiment, we changed the sizes of the processed messages. We used 5 tweets per message (around 500 KB per message). The results presented in Fig. 5 show that Samza and Flink are very efficient compared to Spark, especially for large messages.

4 CONCLUSION

In this work, we surveyed popular frameworks for large-scale data processing. After a brief description of the main paradigms related to Big Data problems, we presented an overview of the Big Data frameworks Hadoop, Spark, Storm and Flink. We presented a theoretical and experimental comparative study of the presented frameworks on a cluster of machines.

REFERENCES

[1] Amir Gandomi and Murtaza Haider. 2015. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management* 35, 2 (2015), 137 – 144.

[2] Wissem Inoubli, Sabeur Aridhi, Haithem Mezni, Mondher Maddouri, and Engelbert Mephu Nguifo. 2018. An experimental survey on big data frameworks. *Future Generation Computer Systems* 86 (2018), 546–564.

[3] A Oguntimilehin and EO Ademola. 2014. A Review of Big Data Management, Benefits and Challenges. *A Review of Big Data Management, Benefits and Challenges* 5, 6 (2014), 433–438.